International Journal of Digital Information and Wireless Communications (IJDIWC) 2(1): 1-6
The Society of Digital Information and Wireless Communications, 2012(ISSN 2225-658X)

Fractals Image Rendering and Compression using GPUs

Munesh Singh Chauhan and Ashish Negi
Information Technology Department
Ibri College of Applied Sciences
Ministry of Higher Education, Sultanate of Oman
munesh.ibr@cas.edu.om

ABSTRACT

Fractal image compression provides immense advantages as compared to conventional image compressions. Though the fractal image encoding time is comparatively quite high as compared to the conventional ones but the decoding time is far less and almost instantaneous. Besides, fractal images are resolution-independent, implying that these images will render the same intensity and quality even when scaled. In other words the number of pixels remains unchanged even while extrapolating the image. In addition to it, the fractal image quality remains un-altered even at low-bit rates thus making it a suitable candidate for offline applications. The present baseline approach for fractal image compression is modified and supported with advanced parallel hardware in the form of Graphical Processor Units from Nvidia Corporation. The GPUs consist of many cores thus providing SIMD parallel processing capability at an un-imaginable rate of around 24 GFLOPS. This processing speed was not possible earlier before the advent of GPUs except in some selected highly evolved supercomputers. The rendering of image and its compression is implemented using OpenCL library. The benefits of faster fractal compression lie in the realm of medical imaging, satellite reconnaissance, gaming & film media.

KEYWORDS

GPU, fractal, attractor, affine transformation, contractive, Iterated Function System (IFS)

1 INTRODUCTION

In a modern society data plays a pivotal role. An effective control, transfer and access of data define the technological advancement in a given society. Images form an important component of data. Since image transmission consumes the bulk of the bandwidth, it becomes imperative to compress images. This will lead to faster transmissions and less costs. Most image compression algorithms work on popular standards such as JPEG [1] (still images), MPEG [2] (motion video images), H.261 [3] (Video telephony on ISDN lines), and H.263 [4] (Video telephony on PSTN lines). All of these algorithms are based on Discrete Cosine transformation [5] (DCT). One of the major drawbacks of most of these standards is that they scale poorly on resolution front. Fractals provide an alternative solution to this problem. Fractals basically denote the self-similarity in image. Fractals can be seen almost everywhere in nature, such as ferns, galaxies, weather, population patterns, coast lines, and stocks to name the few.

2 FRACTALS AND ITS PROPERTIES

The term fractal was coined by Benoit Mandelbrot [6] in 1975. Fractal derives its meaning from the Latin word, “fractus”, meaning fractured. Fractal in

general means “a shape that can be split into parts, each of which is (approximately) a reduced-size copy of the whole”. This property is termed as “self-similarity”. For example, LINUX is a self-recursive acronym that stands for Linux Is Not UniX (Linux Is Not UniX Is Not UniX Is Not UniX…).
Fractal based image formats have not gained usage due to the patent protection and computational intensity of searching self-similar patterns. However decoding can be done quickly which is suitable for video playback.

3 APPLYING FRACTAL IMAGE COMPRESSION

Consider in Figure 1 a special photocopying machine [7] (also termed as MRCM- Multiple Reduction Copying Machine) that while copying reduces the input image by half. Further assume, multiples of input images are iteratively put back into the copying machine with some pre-planned orientations and positions. After a few iterations it is noticed that the image gets reduced to a standard image (called attractor) which in effect cease to change in spite of further iterations. This image is termed as fractal. Additional iterations only lead to high definition and more image clarity. In Figure 2, the attractor remains unchanged even if the initial image is different. Only the position and orientation of the copies determines the final image.

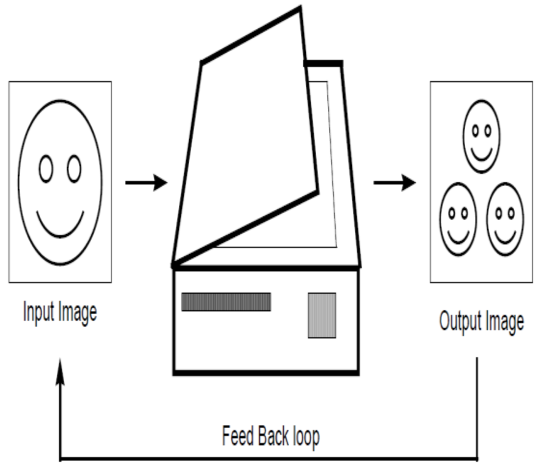

Figure 1: A MRCM that makes three reduced copies of the input image

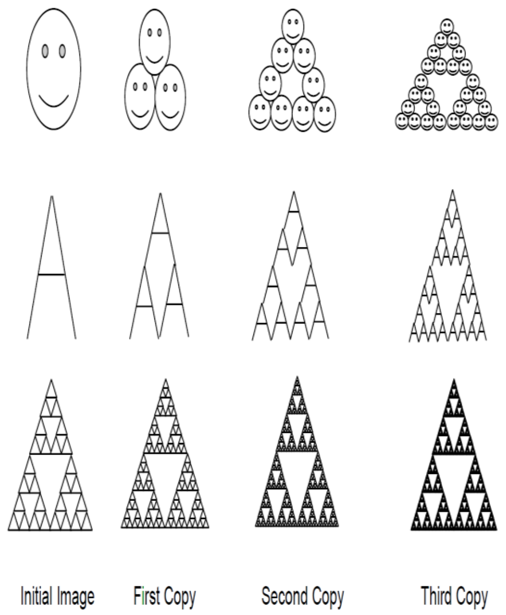

Figure 2: The first three copies of the initial image

Thus, running the copying machine as a feedback loop creates transformations. These transformations can be of different types. Those transformations that are contractive lead to images that can be termed as fractals. The image is said to be contractive if any two points in an input image are found to be closer in a copy.

$$\omega_i \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} e_i \\ f_i \end{bmatrix} \qquad [1]$$

Selecting the transformation in Equation 1 provide good attractors. The transformation shown in Equation 1 is called affine transformation of the plane. These transformations can skew, scale, stretch, rotate, and translate an input image.

M. Barnsley [8] suggested that transformations can be used for image compression. According to him each affine transformation ω_i is defined by six numbers, a_i, b_i, c_i, d_i, e_i and f_i which can be conveniently stored in a computer (6 numbers per transformation × 32 bits per number) whereas storing of an actual image may run into megabytes.

The fractal compression deals primarily with the issue of self-similarity between larger and smaller portions of an image. The original image is partitioned into blocks of fixed size called range and thereby creating a codebook [9] that maps the range with domains consisting of equal sized blocks of the double size of the original image [10].

As shown in Section 2, the process of recursively applying a given definition is termed as Iterated Function System (IFS). In other words, the output of IFS is bound to be self-similar in nature & form. An example to this can be found in Barnsley fern [Figure 3]. The fern is created by iterating four linear equations over a few points. The result is an astonishing image with intense organic details. The reason for the IFS leading to a single image is that these functions are contractive. Though it seems initially incomprehensible how an iterative function contract over a fixed point, but if one looks at the function 1/x that approaches zero when x tends to infinity, it becomes clear that the behavior can be expected. Thus a function can be re-generated using the initial functions.

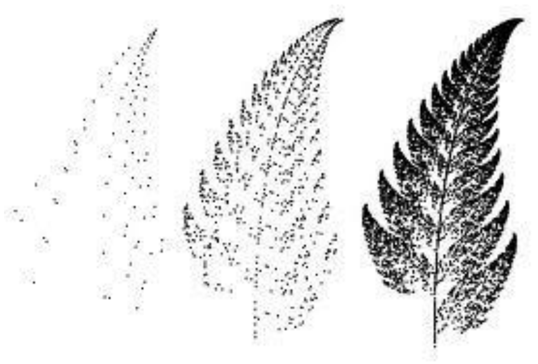

Figure 3: The Barnsley Fern

From the above it is clear that using IFS, image compression is possible if a function can be derived that converge on an image, then that image can be represented entirely in terms of the parameters of a function along with the set of initial conditions [11]. If it is possible to predict as to how each part of an image is similar to another part, then the function can regenerate image by iterating over the initial conditions. This leads to high level image compression but the very bottleneck of searching for function that closely resembles the original image remains.

One of the many approaches to the above mentioned problem is by superimposing a square grid (finer) on an image and looking for functions that closely resembles each cell in the grid. The cells in the grid are termed as the range. We take another coarser grid (twice the size of the finer grid) the cells/ blocks of which are termed as domain. The range block is taken to be 4 × 4 pixel size. The equivalent domain blocks are of 8 × 8 pixel size. The actual compression/ encoding comprises of an IFS that searches for a closest match of a domain block to any one of the range blocks. The comparison of a domain to a range block involves contraction of the domain such that it closely resembles a corresponding range. The collection of contracted domain blocks are put together in a pool called domain pool. This pool creates a set of finite image fragments to search from, thus simplifying the search. Once each domain block is finally transformed to a range block, the resulting transformation is simple and further issues involving brightness and contrast are dealt in a straight forward manner. The brightness in an image block is treated as a mathematical equivalent of multiplying all pixels in a block by a constant number whereas the contrast is dealt with adding a constant to each pixel in a block. So in sum the IFS for a given image consist of contraction, contrast, brightness and translation. Since the mapping of domain to the range block is not absolute the difference/ error is measured as RMS (Root Mean Square) value. The process of finding the RMS involves the difference between the value of the range pixels and the domain pixels. A pixel can take values between 0 and 255.

Thus each range block is identified in the searching process with its corresponding domain block. Finding an optimum set of IFS functions that involve a single domain block that optimally matches each range block comes under the category of NP complete problem [12]. There are many heuristics algorithms proposed but the problem of encoding for a video playback environment still remains unresolved.

The decoding of image is in fact a fairly easy process. This issue will not be delved in detail in the study as it is beyond the purview of this paper.

4 GRAPHICAL PROCESSING UNIT (GPU) CAPABILITIES

Graphics hardware has become specialized and powerful over a period in time. Different mathematical operations on vector matrix can now be computed in a fraction of seconds using GPUs. Even the geometric shapes can now be conveniently drawn. Beyond this, the shade and the contrast issues in images are easily realizable using GPUs [13]. Today the GPUs as a commodity computer chips are among the most powerful and the cheapest per dollar. This has resulted in use of graphics card for scientific computation in almost every compute intensive domain. Since the GPUs are increasingly used for other computations besides graphics, they are often termed as GPGPU (General Purpose Graphical Processing Unit). The most recent GPU is based on the Kepler architecture from Nvidia Corporation^{TM}. It is till date considered the most powerful and the most efficient. The Nivdia GeForce GTX 680 belongs to this family. It is based on 28nm scale fabrication. It took approximately 1.8 million man hours for the development. GTX 680 consists of 192 streaming microprocessors on each core. In total there are 8 cores thus having in total 1536 microprocessors. More than 3.5 billion transistors are packed into this GPU. In terms of performance it is only one of it is only one of its kind that is capable of running High Definition monitors in stereoscopic 3D.

5 EXPERIMENTING WITH ENCODING USING GPU

The CPU as compared to the GPU is suitable for sequential processing. This scheme is suitable when the processes are involved in significant IO (Input Output) or if there is a lot of network traffic. This time delay that results from IO or network activity can be utilized by the CPU to schedule another process. As a result multitasking can be easily implemented. Whereas the GPUs are more suitable for compute intensive tasks that need to be performed on independent data sets. The multiple stream processors can be channelized to work on different data sets and their intermediate results are amortized and finally collated into an output. The prime feature of this scheme is parallelization. The major factor in GPU computing is that the data sets are independent of each other.

During the encoding of the image the search for a function that is optimum for a particular range is independent of the search for function for another range [14]. Since the range blocks comprise of pixels, the parallelization can be implemented at the level of pixels [15]. Similarly the domain contraction can also be achieved in parallel for each domain and their corresponding range.

The experiment involves comparison of fractal compression on a CPU to that of a GPU. The CPU used in question is a T7500 Intel Core 2 Duo processor with each of its core having a speed of 2.2 GHz. The GPU used in the experiment is an Nvidia GeForce GT-540M card having 93 cores.

The encoder is implemented as a C++ program run on the CPU while the parallel version uses OpenCL library on GPU. Figure 4 depicts the encoding time for a standard image encoded over CPU whereas Figure 5 shows the encoding time over GPU.

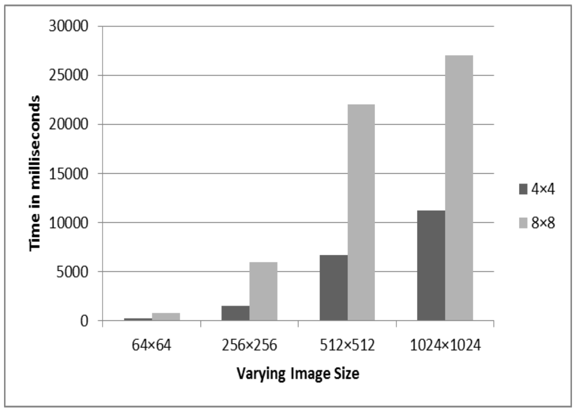

Figure 4: Encoding on the CPU

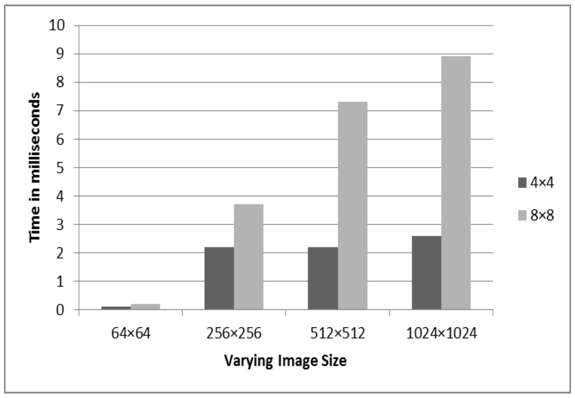

Figure 5: Encoding on the GPU

It can be easily seen that GPU scores over the CPU in the encoding time of an image. Images of different resolutions are considered (64 × 64, 256 × 256, 512 × 512 & 1024 × 1024) and encoded with varying range block sizes (4 × 4, 8 × 8). In all categories GPU time is far lower than the corresponding CPU time. This is basically due to the independence of pixels in a range. Each pixel can be computed in effect independently without any dependence on another.

The encoding time using GPU clearly shows that it is possible to use fractals for resolution independent video capture format. In an ideal high definition (HD) television, the frame rate required for high fidelity display is 16.67 milliseconds. Using this rate the content can be rendered on any type of display panel that supports HDTV. The experiment clearly shows the frame rate below 9 milliseconds which is extremely ideal.

6 CONCLUSIONS AND FUTURE WORK

It is clear that graphical processor units can be used for parallel applications that require intensive computation and have complex algorithmic structures. The graphical units have revolutionized the way scientific computations are done and provide an inexpensive alternative for everyday commodity computing. Comparing the cost to the previous SIMD parallel machines, GPUs have created a more affordable avenue for desktop computing. The use of GPUs is being contemplated in finding self-similarities using IFS in sound waves akin to fractals. The only change being to replace contrast & brightness variables in fractals to that of Fourier transforms for sound waves. Another aspect that is being researched using GPU is the pattern recognition in gene encodings. In short the future of GPU is exciting, and its use in other domains is still being researched.

7 REFERENCES

1. G. K. Wallace, “The JPEG still picture compression standard,” IEEE Trans. Consumer Electronics, vol. 38, no. 1, pp. xviii-xxxiv, Feb. 1992.
2.]MPEG-7 Whitepaer, Sonera MediLab, 13 October, 2003.
3. Line Transmission of Non-Telephone, Signals Video CODEC for AudioVisual Services, ITU-T Recommendation H.261
4. Series H: AudioVisual And Multimedia Systems, Infrastructure of audiovisual services – Coding of moving video, Video coding for low bit rate communication ITU-T, ITU-T Recommendation H.263
5. N. Ahmed, T. Natarajan, Discrete cosine transform, IEEE Transactions on Computers, 1974
6. Mandelbrot, B.B.: The Fractal Geometry of Nature. W.H. Freeman and Company. ISBN 0-7167-1186-9 (1982)
7. J. Kominek, Advances in fractal compression for multimedia applications, Multimedia Systems, Springer-Verlag 1997
8. Barnsley, M.F., Sloan, A.: Chaotic compression. Computer Graphics World (1987)
9. A. Jacquin, Image Coding Based on a Fractal Theory of Iterated Contractive Image Transforms. SPIE Vol. 1360, Visual Communications and Image Processing 1990
10. Owens, J.D. et al: A Survey of General-Purpose Computation on Graphics Hardware. Eurographics 2005, State of the Art Reports, August 2005, pp. 21-51.
11. Sodora, A.: Fractal Image Compression on the Graphics Card. Hopkins Undergraduate Research Journal, Johns Hopkins University (2010)
12. M. Ruhl, H. Hartenstein, Optimal Fractal Coding is NP-Hard, Proceeding, IEEE Computer Society Washington, DC, USA
13. Erra, U.: Toward Real Time Fractal Image Compression Using Graphics Hardware. Advances in Visual Computing, Springer Lecture Notes in Computer Science, 2005, Volume 3804/2005, 723-728, DOI: 10.1007/11595755_92
14. An Introduction to Fractal Image Compression. Literature Number: BPRA065, Texas Instruments Europe, October 1997.
15. Galabov, M.: Fractal Image Compression. International Conference on Computer System and Technologies – CompSysTech’2003.